January 2021

EFFICIENTLY TESTING PART VARIATIONS FOR 3D PRINTING

HP INC

Efficiently testing part variations for 3D printing

Abstract

When refining the part production parameters during 3D printing it is often necessary to print a number of different variations of the same part model. For example, printing many copies at slightly different scale factors to find the optimal production settings.

In this disclosure we describe a new method for automatically packing as many different variations as possible of a part within a given build volume.

There are existing packing systems which can pack as many copies of the same part or different parts as possible in a given volume. Our system differs from these because we pack as many different variants of a single part as possible. In addition, since we are generating the variations, we know how each was generated, and so can automatically generate labels, tables and other aids to help the user later distinguish the different printed variations.

Background

There is a need to print parts in different sizes during testing. The existing way to do that is to generate parts in different sizes, for example using a parameterized CAD application, and then add those different parts to a build for printing. In this case the number of different parts is known in advance by design-of-experiment (DOE), and the procedure for packing/printing is the same as for any set of parts. The scaling factors used are manually recorded and tracked, there is no meta data on what a part was scaled with or when. After the parts have been sintered and processed through metrology the resulting data is also analyzed manually. It is then compared to original scale factors, and a recommended final scale decided on for production.

Thus there is a need to automate the process of generating builds containing multiple copies of different variants (in one example, different sizes) of the same part, for labelling those parts to indicate the variant, and to simplify the subsequent analysis of the printed parts.

Idea

We propose a new solution, where the user specifies a printing Process, two extreme part variations, and additional constraints on the way they should be produced. We then automatically generate and pack a range of part variations, interpolating between those extreme examples.

For example, users may wish to determine the best scaling factor for a part. They then specify the process to be used for producing the parts, and from that we may infer process-specific constraints, for example for our printers it would include the usable volume inside the printer and the inter-part spacing. The user then selects the minimum and maximum scale factors they wish to test. We then automatically generate and pack as many different sizes of the part as possible between those extremes to fill the permissible space while still complying with any constraints. In this way the user receives as many different variants as possible though a simple mechanism. In addition, since we generated the part variants, we may also automatically label the parts to aid with later analysis.

Algorithm:

We're describing a simple case here to start, with more-complex options listed later:

1. User selects a particular production Process from which the system determines the available print volume and other process-specific constraints like the required minimum spacing between parts
2. User selects a Part which they would like to test print at different sizes, along with the minimum and maximum sizes, for example a minimum of 100% and a maximum of 110%.
3. User selects the portion of the print volume which should be used for the test
4. System generates and packs as many different variants of the part as possible between the min/max part sizes to fill the permissible space while still allowing suitable part spacing and complying with any additional Process-specific constraints. For example, parts in the build might be scaled at 100%, 102%, 104%, 108%, and 110% giving 5 different variants if the part was relatively large compared to the available space. But if the part were smaller, then there may be space for more copies at 100%, 101%, 102%, …110%. In this way we provide the user with as many options as possible given their constraints.

   To determine how many parts fit there are many options. Here we provide a simple and rather brute-force example:

   1. Find a quick and very-conservative lower-bound on the number of parts. For example, by taking the size of the larger part bounding box, expanding by the inter-part spacing, and then dividing the volume of the available space by the volume of that box
   2. Find a quick and very-conservative upper bound on the number of parts. For example, by dividing the volume of the available space by the volume of the smallest part
   3. Now use a binary search to find the largest integer value between those two extremes where the variants all still fit. To test if parts fit, we can use an existing packing algorithm.

5. (Optionally) generate a label for each part, that label indicating the scale factor, or an index/key into a table of scale factors. The label may be as simple as a single character, or as complex as a QR code. Apply the label, either to each part, or on a tag attached to each part, or on a disconnected but inseparable

item (for example a cage or attached tag), or on a plaque printed nearby the part so they may be kept together during subsequent processing. And then if the label is to a table of scale factors, then store, display, or print the table for the user. In one example the table itself may be 3D-printed along with the parts.

The possible choices will be process specific. For example, the fragility of parts produced with some processes may preclude adding tags or embossing labels. Even then, some parts may be too small to have more than one or two characters, so a key may be required.

In other examples we may allow the user to specify other or additional constraints, for example using cost, time, or material usage:

"I want to test this part in as many different sizes as possible between 95% and 105%, while using no more than 10 kg of green material to produce the test parts.

In other examples we may have a more interactive system, where the user may be asked:

"Would you like to print 3 different sizes consuming 1/4 of the build, 4 different sizes consuming 1/3 of the build, or 6 different sizes using 1/2 of the build.

In other examples, combinations of options are possible:

"if we can fit 100 parts in a build we need to decide if we want 10 parts across 10 scale factors, or 20 parts across 5 scale factors."

In another example the solution may look like a decision tree software, where you first define your print area and part. It displays what it can fit in the build with the orientation constrained, then you decide how many parts per scale factor, then the data range.

In another example the part variations may be more complex than scaling the whole part in all dimensions. In that case the user may provide two parts one for each extreme, and we would interpolate between those. For example, the user may have one part which is narrow and another wide, and we would then vary the part width rather than scaling all dimensions of the part.

In another example the user may provide (or interact with the system to edit) a mapping between the parameters to be varied and the generated parts. For example, the user may prefer a logarithmic mapping where instead of the parts being: 100, 102.5, 105, 107.5, 110, they are instead 100, 101, 102, 105, 110

Advantages

The user doesn't have to choose in advance the number of different variations to test. Instead they give the constraints and the system automatically produces as many variations as possible under those constraints.

There are existing systems which can pack as many copies of a part as possible, but our system is different because we are packing as many different variations of a part as possible.

There are also systems which can generate interpolated models between two examples. The distinction in our case is that we determine the number of variations based on the number which can be packed in the space allocated by the user according to their supplied constraints.

In addition, since we are generating the variations, we know how each was generated, and so can automatically generate labels, tables and other aids to help the user distinguish the different printed variations.

Disclosed by Benjamin Howorth and Craig Peter Sayers, HP Inc.